LOAN DOCUMENT

PHOTOGRAPH THIS SHEET

AD-A276 363

DTIC ACCESSION NUMBER

LEVEL

INVENTORY

(1)

WL-TR-94-1009

DOCUMENT IDENTIFICATION

Jan 94

Approved for public release
Distribution Unlimited

DISTRIBUTION STATEMENT

ACCESSION FOR

NTIS GRA&I ☑
DTIC TRAC ☐
UNANNOUNCED ☐
JUSTIFICATION

BY

DISTRIBUTION/

AVAILABILITY CODES

DISTRIBUTION	AVAILABILITY AND/OR SPECIAL	
A-1		

DISTRIBUTION STAMP

DTIC
ELECTE
MAR 1 1994
S C D

DATE ACCESSIONED

DATE RETURNED

94 2 24 1 5

DATE RECEIVED IN DTIC

94-06192

REGISTERED OR CERTIFIED NUMBER

HANDLE WITH CARE

PHOTOGRAPH THIS SHEET AND RETURN TO DTIC-FDAC

Best
Available
Copy

WL-TR-94-1009

THE MAINTENANCE OF OPERATIONAL FLIGHT PROGRAM

CHARLES P. SATTERTHWAITE

JANUARY 1994

FINAL REPORT FOR 10/05/92-10/08/92

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS *UNLIMITED*.

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7409

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely Government-related procurement, the United States Government incurs no responsibility or any obligation whatsoever. The fact that the government may have formulated or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication, or otherwise in any manner construed, as licensing the holder, or any other person or corporation; or as conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

This report is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and is approved for publication.

Charles P. Satterthwaite
Project Engineer
WL/AAAF-3

Robert L. Harris
Chief, WL/AAAF-3

Donna M Morris
Chief, WL/AAAF

If your address has changed, if you wish to be removed from our mailing list, or if the addressee is no longer employed by your organization please notify WL/AAAF, WPAFB, OH 45433-6543 to help us maintain a current mailing list.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

JAN 1994 FINAL 10/05/92--10/08/92

THE MAINTENANCE OF OPERATIONAL FLIGHT PROGRAM

C
PE
PR 9994
TA 00
WU 00

CHARLES P. SATTERTHWAITE

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7409

AVIONICS DIRECTORATE
WRIGHT LABORATORY
AIR FORCE MATERIEL COMMAND
WRIGHT PATTERSON AFB OH 45433-7409

WL-TR-94-1009

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

THE PROCESS OF MAINTAINING OPERATIONAL FLIGHT PROGRAMS (OFPS) IS DISCUSSED SO THAT INTERESTED INDIVIDUALS CAN UNDERSTAND (1) HOW OFPS WORK, (2) HOW OFPS ARE CHANGED, (3) HOW OFPS ARE TESTED, (4) HOW OFPS ARE DOCUMENTED, (5) HOW TO TRAIN OFP MAINTAINERS, AND (6) HOW TO MEASURE OFPS.

7

UNCLASSIFIED UNCLASSIFIED UNCLASSIFIED UL

THE MAINTENANCE OF OPERATIONAL FLIGHT PROGRAMS

Charles P. Satterthwaite

Avionics Logistics Branch
Wright Laboratory
Wright Patterson AFB OH 45433-6543

Abstract

The process of maintaining Operational Flight Programs (OFPs) is discussed so that interested individuals can understand (1) how OFPs work, (2) how OFPs are changed, (3) how OFPs are tested, (4) how OFPs are documented, (5) how to train OFP maintainers, and (6) how to measure OFPs.

Summary

Embedded computers are increasingly called upon to provide high-tech solutions to complex multiple threat type environments for today's generation of weapon systems. The heart of an embedded computer is its software, which is the Operational Flight Program (OFP). In understanding the role of an OFP, one must thoroughly understand the interaction of an OFP in its system, the processes associated with changing OFPs, the structure of an OFP, the weapon system and mission requirements of an OFP, the OFP's support environment, the testing of OFPs, the documentation of OFPs, the training of OFP maintainers, and the metrics of OFPs. This paper addresses each of these issues.

How Does An OFP Work?

The Operational Flight Program (OFP) literally is the software portion of an embedded computer system. The computer and its periphery interfaces make up the system hardware. The hardware enabled by the OFP software describes the whole system. The embedded computer system has partitioned memory which is filled with some type of machine level (binary) code. The OFP is loaded into this partitioned memory and, when enabled, empowers the whole system to perform its desired functions. Each embedded computer system has an instruction set which is burned into its Read Only Memory (ROM). The instruction set allows the embedded computer maintainer access and the capability to optimize the remaining partitioned memory. The level of sophistication of a embedded computer system is described by its instruction set, its memory, and its throughput [1,2,4].

What Drives Changes To An OFP?

Given a working OFP in a working system, why would I ever want to make changes? One reason would be that the users of the system would require an altered mission. An example of this would be the Tactical Air Command (TAC) (now Air Combat Command ACC) requesting an Engineering Computer Change. A typical TAC Form 37 would be a request to provide a clearer display for the pilot under some given condition.

Another reason would be that some flaw is discovered while the embedded computer system is operational. Some combination of events might cause partial or total system failure, prompting a review and redesign in the affected areas of hardware, software, or both [4].

How Is An OFP Changed?

Diagnosis/Analysis/Isolation/Integration/Test

Given the task of changing an OFP (making a new version or even a new block cycle), several steps are followed to bring about the change. First, the requested change(s) is/are diagnosed so that their purpose is understood. Engineers and pilots don't always view life in parallel, so careful review keeps the OFP maintainer on track. Once the OFP maintainer thoroughly understands the change request, he makes an analysis as to which OFP areas he must alter. Usually the OFP is made up of a series of modules with specialized functions which will be covered in more detail later. A typical TAC Form 37 change might impact three modules of a forty module OFP. The OFP maintainer will next isolate these modules by making copies of them and implementing his design changes to his copies. The OFP maintainer integrates his assembled modules by linking it together with the other unaltered modules to form his own unique OFP. The OFP maintainer's final task is to test out his OFP by putting it through an acceptance test procedure, which wrings out the new OFP. For a sizable OFP with significant TAC Form 37 change requests, several maintainers would follow these procedures simultaneously, and then a lead maintainer integrates and tests the new OFP [4].

The Modules/Functions Of The OFP

Many OFPs are made up of modules which partition the OFP into its functions and sub-functions. A typical Fire-Control Computer contains air-to-air, air-to-ground, navigation, control and display, executive, Heads-Up-Display (HUD), and over-load warning functions, each of which has one or more sub-functions. An example of a sub-functional module would be a air-to-air 50 cycles per second module. The air-to-air function might be made up of three modules (10/sec, 20/sec, and 50/sec). Many of the modules would have inter-dependencies. For example the executive modules would determine the timing and priority scheduling among the entire OFP [4].

The Weapon System/Mission

In order to make OFP changes, a maintainer must understand the weapon system for which his embedded computer is a part, and the mission for which that weapon system is required. Many times the availability for new functions in a embedded computer system are limited, so that a trade-off analysis must be performed in order to optimize the mission and the weapon system. A sub-function which is rarely or never utilized might be sacrificed in order to accommodate a new requirement of higher priority to TAC [1,2,4].

The Support Environment

In order to maintain an OFP, the maintainers require a dedicated computer system and a simulation environment. The dedicated computer system allows the maintainer to access OFPs as well as copy and alter OFPs as required. The simulation

environment allows maintainers to run their OFPs enabling them to debug and test interactively. The hardware of a dedicated computer system usually includes main-frame computers (or powerful engineering workstations), various types of printers, various disk storage devices, networking, and several access terminals. An example used by the F-15 Central Computer OFP Maintainers is the Harris Operating System with Harris 800 and 1200 Mainframes, as well as a complimentary host of Harris Printers, Disk Drives, and Reel to Reel Drives [3,4].

How Is An OFP Tested?

The ultimate test of an OFP is that it becomes the operational version. But several layers of testing exist before OFPs are accepted. Flight tests are expensive, as are full-up simulations. But some confidence can be gained through wringing the OFP out on its software simulated environment. The process which wrings an OFP out is called the acceptance test procedure (ATP). Various other tests are required in the software development life cycle of OFPs. These include tests of the target processor (and its environment), peculiar tests, and the Operational Test and Evaluation (OT&E) [4].

The Acceptance Test

The OFP maintainers primary test is the acceptance test procedure (ATP). This test is designed to wring out an OFP to a degree that it can be released with confidence to flight test and then operational test and evaluation. The ATP is a chronological check of the OFP's responses to inputs. Inputs include switch positioning, preset conditions such as altitude or airspeed, and hardware interrupts to name a few. The OFP loaded into its embedded computer and hosted on its simulation environment responds to these inputs in the form of static or dynamic displays, which can be checked against expected results [4].

Automated Tests

As the complexity of OFPs increases with software usage, the ability to manually perform acceptance test procedures (ATPs) decreases or the ability to fully test OFPs decreases. The F-15 Central Computer OFP Acceptance Test currently takes two man weeks. Much of this F-15 ATP is static testing in which the maintainer is flipping switches and verifying displays. This ATP time requirement for manual check-out will soon be man hour prohibitive for new versions of OFPs with orders of magnitude more code. One possible solution is to automate as much of the ATP as possible by utilizing the shared memory and remote control features of software engineering work stations. One possible means of implementing this automation is through a tool developed at Wright Laboratory called Automatic Validation (AUTO-VAL) [1,2,3,4].

Iterative Nature Of OFP Testing

Usually OFPs are not acceptable in their first cut, even when they go through OT&E. Five or six cycles through the testing process is not unusual. Much of this is related to the complex nature of OFPs, poor pilot-to-engineer feed-back-loops, and changing mission requirements midstream in OFP development [4].

How Is An OFP Documented?

Several types of documentation exist to support the development and maintenance of OFPs. Technical Orders (TOs) are most prevalent with Version Description Documents (VDDs) being most common. Documentation such as the Technical Description Document (TDD), TAC Form 37 Engineering Change Requests also exist, plus a host of ancillary notes generated when development work occurs. Most documentation occurs after an OFP is wrung out. The lead maintainer writes a synopsis of changes made between versions, which gets interpreted into the VDD and the TDD. TOs usually follow several months after an OFP checks out. Proper documentation allows each level of the OFP software development cycle to be visible and specified to the level of detail required [1,2,3,4].

Automated Documentation

The iterative nature of maintaining Operational Flight Programs tends to cause the documentation process to occur as a final step, rather than with each iteration. This causes much valuable information to be lost. The capture of mistakes is necessary because you know what not to do. Abandoned efforts might be called upon in future OFP change activities. Unfortunately, documenting changes is tedious work so it is put off as a last phase effort. Most of the interim information gleaned in development is lost. Documentation tools could be built into the maintainers toolbox so that whenever he assembled his source code, some minimum set of documentation would be recorded. This tool could capture the user, date and time, files altered, and prompt for explanations of additions, deletes, new variables, and logic flow [1,2,3,4].

How Do You Train Maintainers?

The training of OFP maintainers requires multiple levels of instruction which include weapon system, target processor, dedicated computer system, simulation system, and integrated testing, plus the facility requirements such as security that the new maintainer must learn. Often the new man is on his own, without a proven method or mentor to bring him up to speed [4].

Training - The Weapon System/ Mission And Major Components

It is important to keep in perspective the reason why your OFP support organization is in existence. The OFP is an integral part of a specific weapon system which has a specific mission. Also of significance are the major components of the weapon system. frequently this perspective is clouded because the OFP maintainer's training program has not been established as an integral part of the OFP software life cycle for that particular weapon system. Also important is that OFP maintainers have a working knowledge of their weapon systems. This knowledge should include the features of the weapon system, the mission of the weapon system, and the associated sub-systems or components of the weapon system. The features of a weapon system include its physical make-up, its capabilities, its crew, and its history. The mission of a weapon system is how the system is being, and will continue to be, utilized. The major components of the weapon system could include its radar, its electronic warfare systems, its armament, and its

communication and navigation systems. Without a continuously updated knowledge of these features of his environment, an OFP maintainer is limited in the scope of his ability to support the weapon system's OFP [4].

Training - Diagnosis/Analysis/Isolation/Integration/Testing

Complex skills required to maintain OFPs are the diagnosis of problems and change requests, the analysis of the resources required to make a change, the isolation of faulty software logic, the integration of multiple software changes, and the design and implementation of detailed OFP testing. Given a clearly stated requirement for OFP change, maintainers have to know how to implement that change and what resources are required to enable their implementation including memory, man hours, integration time, and testing time. A simple OFP change might be a change closely related to a past change, and thus easily performed. A complex change might require that the maintainer obtain specific training, alter large amounts of code, design specialty test scenarios, and spend many hours integrating and debugging the change [4].

How Do We Measure OFPs?

What metrics would help OFP maintainers and managers better understand the cost and complexity of their tasks? Usually the most quoted metric is "Lines of Code". Lines of Code does not account for more efficient coding or coding conventional type changes. Attention needs to be paid to broadening traditional metrics such as Lines-of-Code by discussing other software support parameters such as OFP comparative analysis, software developmental research, maintainer skill level, software quality, and software reuse [4,5].

Comparative Analysis

Comparative analysis is the process by which two or more similar software files are compared to discover the overlapping of the files. Non-overlapping code would be inserted code or deleted code. Further analysis might reveal added or deleted variables, documentation, or even unique modules. A manual comparative analysis is performed by examining the two or more files next to each line-by-line. Identical lines are marked off as such and differences are noted as well. Comparative analysis utilities also exist in software form. These utilities vary in complexity and performance, but essentially automate the manual line-by-line analysis [4,5].

Structured Programming

Structured Programming aligns source code in easy-to-read and digest modules in a top-to-bottom configuration or a bottom- to-top configuration. The modules are designed to perform related tasks and use related variables. A well-structured module contains 50-100 or less lines of code. Modules are duplicated rather than called or sent to, as in the case of a FORTRAN GOTO statement. This could increase the coding effort and memory required, but drastically decreases the code complexity [4,5].

Software Cost Analysis

The OFP maintainer is increasingly called upon to identify the costs related to each phase of his OFP

software development. The problem with this is that many software projects and resources overlap. A test plan or a complex algorithm might be used over and over again with slight modification. How do you attribute the original high overheads to later projects? What value is placed on the skill level of the individual OFP maintainers? A senior engineer with an intimate knowledge of a complex system should be considered an invaluable asset. In order to truly represent software costs, values have to be placed on the individual resources and processes used throughout a OFP's Life Cycle Development. These resources and processes must be carefully differentiated between OFP block cycles to properly allocate their individual project value [4].

References

[1] Harris,R.L.,"Laboratory Concepts in Avionics", IEEE/AIAA/NASA 9th Digital Avionics Systems Conference Proceedings, 15-18 October 1990, Virginia Beach VA.

[2] Morris,D.M.,"Avionics Operational Flight Program Supportability", IEEE/AIAA/NASA 9th Digital Avionics Systems Conference Proceedings, 15-18 October 1990, Virginia Beach VA.

[3] Satterthwaite,C.P.,"Getting a Handle On Designing For Avionics Software Supportability and Maintainability", IEEE/AIAA National Aerospace Electronics Conference Proceedings, 18-22 May 1992, Dayton OH.

[4] Satterthwaite,C.P., *Maintaining An Operational Flight Program (OFP)*, Air Force Technical Memorandum WL-TM-91-123, Wright Patterson AFB,OH,1991.

[5] Tso,K.S.,Hecht,M., Littlejohn,K.,"Complexity Metrics For Avionics Software", IEEE/AIAA National Aerospace Electronics Conference Proceedings, 18-22 May 1992, Dayton OH.